Security Board

BNX-C206

Always at the forefront of innovation

User Manual

Copyright

This publication contains information that is protected by copyright. No part of it may be reproduced in any form or by any means or used to make any transformation adaptation without the prior written permission from the copyright holders.

This publication is provided for informational purposes only. The manufacturer makes no representations or warranties with respect to the contents or use of this manual and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. The user will assume the entire risk of the use or the results of the use of this document. Further, the manufacturer reserves the right to revise this publication and make changes to its contents at any time, without obligation to notify any person or entity of such revisions or changes.

© 2011. All Rights Reserved.

Trademarks

All trademarks and registered trademarks of products appearing in this manual are the properties of their respective holders.

FCC and DOC Statement on Class A

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a residential installation. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and the receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio TV technician for help.

Notice:

1. The changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.
2. Shielded interface cables must be used in order to comply with the emission limits.

Warranty

1. Warranty does not cover damages or failures that are raised from misuse of the product, inability to use the product, unauthorized replacement or alteration of components and product specifications.
2. The warranty is void if the product has been subject to physical abuse, improper installation, modification, accidents or unauthorized repair of the product.
3. Unless otherwise instructed in this user's manual, the user may not, under any circumstances, attempt to perform service, adjustments or repairs on the product, whether in or out of warranty. It must be returned to the purchase point, factory or authorized service agency for all such work.
4. We will not be liable for any indirect, special, incidental or consequential damages to the product that has been modified or altered.

Static Electricity Precautions

It is quite easy to inadvertently damage your PC, system board, components or devices even before installing them in your system unit. Static electrical discharge can damage computer components without causing any signs of physical damage. You must take extra care in handling them to ensure against electrostatic build-up.

1. To prevent electrostatic build-up, leave the system board in its anti-static bag until you are ready to install it.
2. Wear an antistatic wrist strap.
3. Do all preparation work on a static-free surface.
4. Hold the device only by its edges. Be careful not to touch any of the components, contacts or connections.
5. Avoid touching the pins or contacts on all modules and connectors. Hold modules or connectors by their ends.

Important:

Electrostatic discharge (ESD) can damage your processor, disk drive and other components. Perform the upgrade instruction procedures described at an ESD workstation only. If such a station is not available, you can provide some ESD protection by wearing an antistatic wrist strap and attaching it to a metal part of the system chassis. If a wrist strap is unavailable, establish and maintain contact with the system chassis throughout any procedures requiring ESD protection.

Safety Measures

To avoid damage to the system:
• Use the correct AC input voltage range.

To reduce the risk of electric shock:
• Unplug the power cord before removing the system chassis cover for installation or servicing. After installation or servicing, cover the system chassis before plugging the power cord.

Battery:
• Danger of explosion if battery incorrectly replaced.
• Replace only with the same or equivalent type recommend by the manufacturer.
• Dispose of used batteries according to local ordinance.

Before Using the System

Before using the system, prepare basic system components.

If the system comes as a barebone; that is, none of the key components, including processor, memory, and hard drive has been pre-installed as part of your purchase, you will need to at least ensure a compatible counterpart is located and installed.

You will also need a few external system peripherals intended for the use of the system, a common pool with at least a keyboard, a mouse, and a monitor is thus suggested.

Table of Content

Copyright 2
Trademarks 2
FCC and DOC Statement On Class A 2
Warranty 3
Static Electricity Precautions 3
Safety Measures 4
Before Using the System Board 4
Table of Content 5

Chapter 1 General Information
1.1 Main Feature 8
1.2 Specifications 10
1.3 Optional LAN Modules 11
1.4 Board Layout 12

Chapter 2 Jumper Setting
2.1 Before You Begin 14
2.2 Precautions 14
2.3 Setting Jumpers 15
2.4 Location of Jumpers and Connectors 16
2.5 Jumpers 18
2.6 Internal Connectors 19

Chapter 3 Operation
3.1 System Memory 24
3.2 Installing Memory 24
3.3 Installing LGA1155 Intel® Core-i CPU, Heat Sink, and Fan 26
3.4 Adding Power Connectors 29
3.5 Adding CompactFlash Cards 30
3.6 Adding miniPCI Cards 31

Chapter 4 BIOS Setup
4.1 Entering Setup 34
4.2 Getting Help 34
4.3 Control Keys 34
4.4 The Main Menu 35
4.5 The Advanced Menu 36
4.6 The Chipset Menu 39
4.7 The Boot Menu 40
4.8 The Security Menu 41
4.9 The Save & Exit Menu 42
4.10 Event Logs 44

Chapter 5 Programming Guide

5.1 LAN Status Sample Code 46
5.2 LAN Switch Sample Code 51
5.3 WDT Sample Code 60

Chapter 1

General Information

1.1 Main Feature

Network Security Board

BNX-C206 is a Network Security Board, featuring on Intel® C206 chipset, supports LGA1155 Intel® Xeon E3-1200/1200V2, and Gen-2/3 Celeron®, Pentium®, Core®-i3/i5/i7 processors that carry the built-in Intel® HD Graphic engine. Below is a brief list of some compatible processors as a quick reference:

- Celeron® Processor: G540/G1620
- Pentium® Processor: G850/G2120
- Core®-i3 Processor: i3-2120/i3-3220
- Core®-i5 Processor: i5-2400/i5-3550S
- Core®-i7 Processor: i7-2600/i7-3770
- E3-1200 Xeon® Processor: E3-1225, E3-1225 V2, E3-1275, E3-1275 V2

8GB Memory for 64bit OS

The four Dual Channel DDR3 DIMM slots are designed to carry up to 32GB DDR3 1066/1333/1600MHz SDRAM, ideally facilitating applications that demand total memory capacity for the use of 64bit OS, beyond the 4GB barrier inherent in the 32bit OS. With Intel® Desktop processors, only non-ECC RAM modules are supported. However, with Intel® E3-1200 Xeon processors, both non-ECC and ECC RAMs are supported.

Basic I/O Modules

There is no edge connector on BNX-C206 security board. Instead, an IO module can be added to support LAN ports, console port, USB port, and LED indications, etc. Two IO modules are available as:

Module Name	LED	Console	USB	LAN
IO-1	Power LED HDD LED Status LED	1 RJ45	2 Type-A	7 RJ45 Gigabit (82574L)
IO-2	Power LED HDD LED Status LED	1 RJ45	2 Type-A	3 RJ45 Gigabit (82574L) 2 SFP Gigabit (82576EB)

Module Gigabit LAN Ports

The two PCIe X8 expansion slots are designed to make the unit expandable with a variety of LAN modules. Below is a list of all available LAN modules:

- 4-Port RJ45 Gigabit LAN Module on Intel® 82580EB
- 4-Port SFP Gigabit LAN Module on Intel® 82580EB
- 2-Port SFP+ 10G LAN Module on Intel® 82599ES

List of Key Features

- Intel® C206 Chipset
- Intel® LGA1155 E3-1200 and E3-1200 V2 Series Xeon® Processor
- Intel® LGA1155 Gen-2/3 Desktop Celeron®, Pentium®, Core®-i3/i5/i7 Processor
- Four DDR3 RAM Slots up to 32GB, both ECC and non-ECC (Condition applies)
- Four SATA2 + Two SATA3 Ports
- One CompactFlash Socket
- One miniPCI Slot
- 16-bit GPIO Pins
- One Expansion Slot for IO module
- Two Expansion Slot for up to additional 4-Port Gigabit LAN Module or 2-port 10G LAN Module

1.2 Specifications

Core Engine	Chipset	Intel® C206 PCH
	Processor	Support Single Intel® E3-1200/1200 V2 Processor Support Intel® Gen-2/3 Core i3, i5, i7, Pentium®, Celeron® Processor Single LGA1155 Processor Socket
	Memory	4x DDR3 1066/1333 DIMM Slots, up to 32GB Support Non-ECC/Non-Buffered Memory Module (for Intel® Desktop Processor) Support ECC/Non-Buffered Memory Module (for Intel® Xeon Processor Only)
	Display	CPU Integrated
Ethernet	Controller	Onboard 3x Intel® 82574L + 2x Intel® 82576EB GbE Controllers
	LAN Module	Support 2x PCIe X8 Slots for LAN Module
Storage	SATA	2x SATA3 + 4x SATA2 Ports
	CF	1x CompactFlash Socket
	RAID	Support Intel® RAID-0, 1, 5 RAID Function
Internal I/O	VGA	1x Internal DB15 VGA Connector
	PS/2	1x Keyboard/Mouse Pin Header
	COM	1x COM2 Pin Header
	LPT	1x Parallel Box Header
	USB	4x USB2.0 Ports
	GPIO	16x GPIO Pin Header
	Chassis Intrusion	1x 2-pin Pin Header
Front I/O	Indication	1x Power LED, 1x HDD LED, 1x Status LED
	LAN	3x RJ45 1G Ports, 2x SFP 1G Ports
	Console	1x RJ45 Type Console Port
	USB	2x USB 2.0 Ports
Expansion	miniPCI	1x miniPCI Slot
	LAN Module	2x LAN Modules
Other	H/W Monitoring	Monitor temperature, voltage, and fan speed, auto-throttling control at CPU overheat
Environment	Operating Temp.	0ºC ~ 60ºC
	Storage Temp.	-20ºC ~ 70ºC
	Humidity	10% ~ 90% (Non-Condensing)
Mechanical	Dimension	303mm (W) x 230mm (D) x 1.6mm (H)

1.3 Optional LAN Modules

BEM-C600-580-C4	Type	LAN Module
	Chipset	Intel® 82580EB
	Interface	PCIe X4
	Network Port	Four RJ45 Gigabit
	Bypass	1-Pair
BEM-C600-580-F4	Type	LAN Module
	Chipset	Intel® 82580EB
	Interface	PCIe X4
	Network Port	Four SFP Fiber Gigabit
BEM-C600-599-F2	Type	LAN Module
	Chipset	Intel® 82599ES
	Interface	PCIe X8
	Network Port	Two SFP+ 10G

1.4 Board Layout

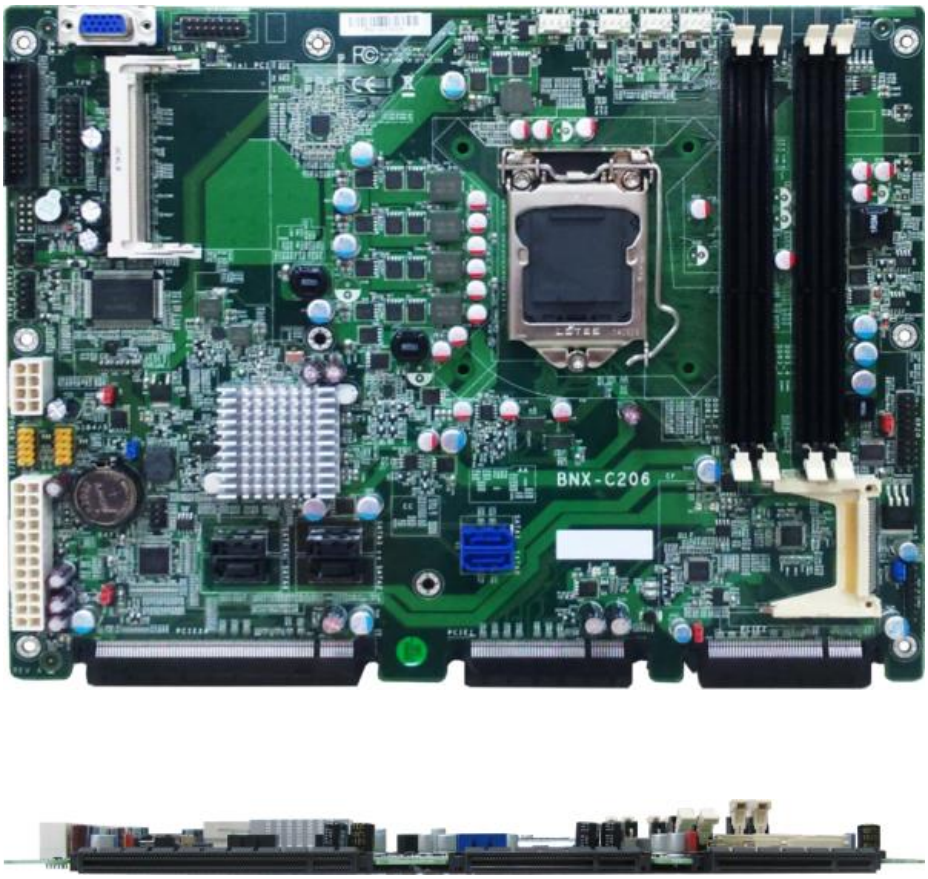


Figure 1.1: Board Layout of BNX-C206

Chapter 2

Preparation

2.1 Before You Begin

A stable and clean working environment are essential. Dust and dirt can get into components and cause a malfunction. Use containers to keep small components separated.
Adequate lighting and proper tools can prevent you from accidentally damaging the internal components. Most of the procedures that follow require only a few simple tools, including the following:

- A Philips screwdriver
- A flat-tipped screwdriver
- A set of jewelers Screwdrivers
- A grounding strap
- An anti-static pad

Using your fingers can disconnect most of the connections. It is recommended that you do not use needle-nosed pliers to disconnect connections as these can damage the soft metal or plastic parts of the connectors.

Before working on internal components, make sure that the power is off. Ground yourself before touching any internal components, by touching a metal object. Static electricity can damage many of the electronic components. Humid environment tend to have less static electricity than dry environments. A grounding strap is warranted whenever danger of static electricity exists.

2.2 Precautions

Computer components and electronic circuit boards can be damaged by discharges of static electricity. Working on the computers that are still connected to a power supply can be extremely dangerous. Follow the guidelines below to avoid damage to your computer or yourself:

- Always disconnect the unit from the power outlet whenever you are working inside the case.
- If possible, wear a grounded wrist strap when you are working inside the computer case. Alternatively, discharge any static electricity by touching the bare metal chassis of the unit case, or the bare metal body of any other grounded appliance.
- Hold electronic circuit boards by the edges only. Never touch the components on the board unless it is necessary to do so. Do not flex or stress the circuit board.
- Leave all components inside the static-proof packaging that they shipped with until they are ready for installation.
- Use correct screws and do not over tighten screws.

2.3 Setting Jumpers

A jumper is the simplest kind of electric switch. It consists of two metal pins and a cap. When setting the jumpers, ensure that the jumper caps are placed on the correct pins. When the jumper cap is placed on both pins, the jumper is SHORT. If you remove the jumper cap, or place the jumper cap on just one pin, the jumper is OPEN. Please see the following illustrations

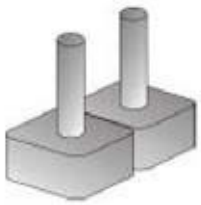
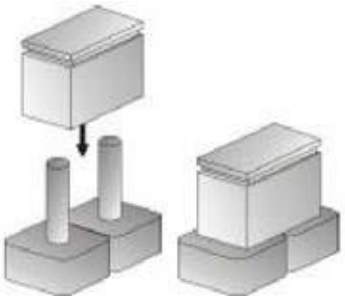
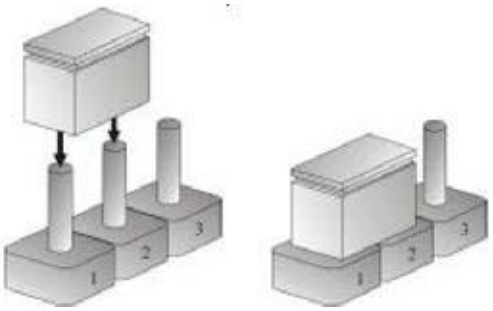
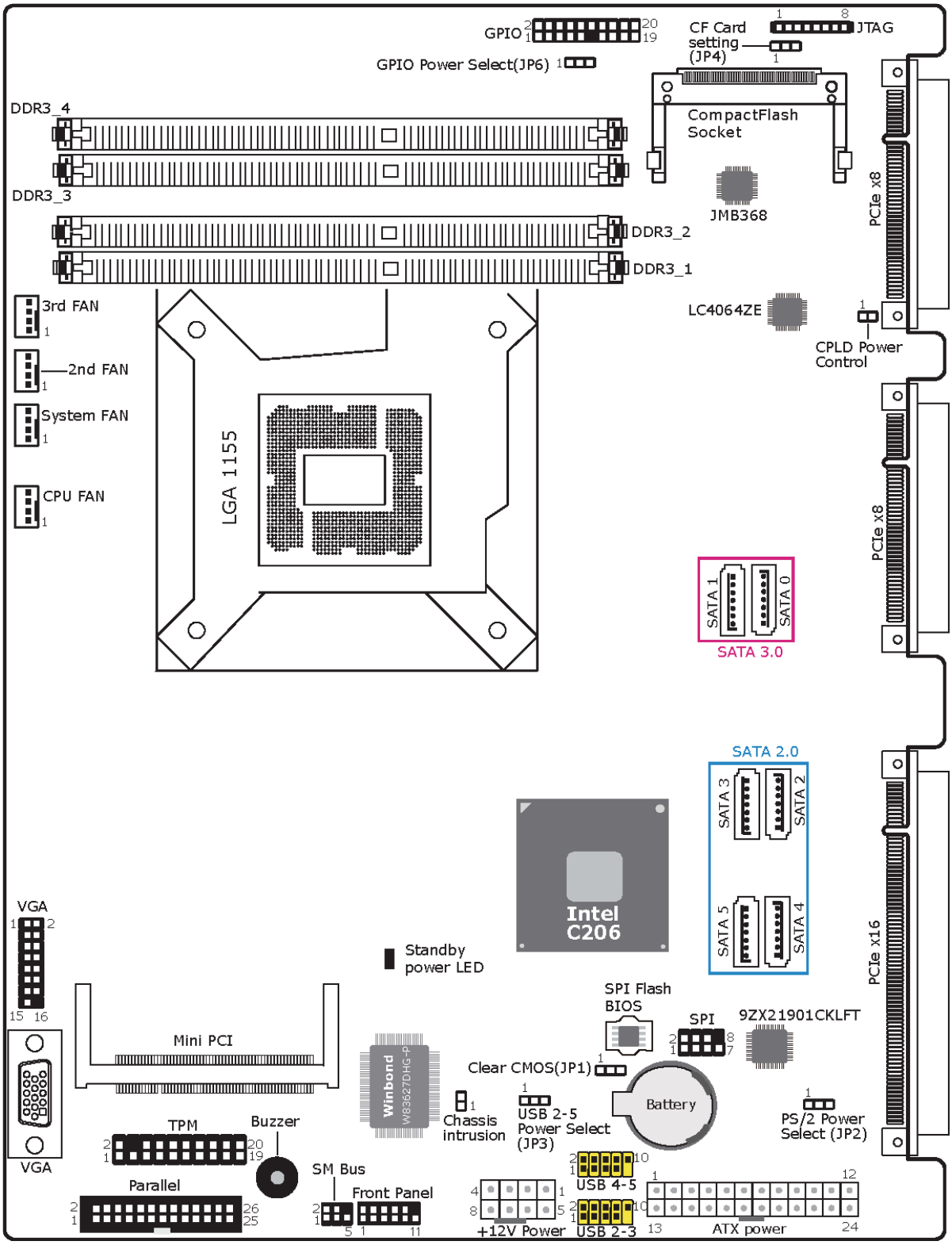
<table>
<tr><td rowspan="2">The illustrations on the right show a 2-pin jumper. When the jumper cap is placed on both pins, the jumper is SHORT. If you remove the jumper cap, or place the jumper cap on just one pin, the jumper is OPEN.</td><td>  </td><td>  </td></tr>
<tr><td>Open (Off)</td><td>Short (On)</td></tr>
<tr><td>These illustrations show a 3-pin jumper. Pins 1 and 2 are SHORT.</td><td colspan="2">  </td></tr>
</table>

Table 2-1: Setting Jumpers

2.4 Locations Of Jumpers and Connectors



List of Onboard Connectors

1	Clear CMOS Jumper JP1
2	PS/2 Power Source Jumper JP2
3	USB2 ~ USB5 Power Source Jumper JP3
4	CF Card Master/Slave Selection Jumper JP4
5	GPIO Pin Power Selection Jumper JP6
6	VGA Connector
7	Parallel Port Connector
8	Front Panel Connector
9	SMBUS Connector
10	USB2-3/ USB4-5 Connector
11	SPI Connector
12	SATA Connector
13	Chassis Intrusion Connector
14	CPU Fan / SYS Fan Connector
15	8+8 GPIO Pin Connector
16	LAN CPLD JTAG Connector

2.5 Jumpers

▶ Clear CMOS Jumper (JP1)

Pin	Definition
1-2	Normal Operation (default)
2-3	Clear CMOS

▶ PS/2 Power Source Jumper (JP2)

Pin	Definition
1-2	5V (default)
2-3	5V Standby

▶ USB2 ~ USB5 Power Source Jumper (JP3)

Pin	Definition
1-2	5V (default)
2-3	5V Standby

▶ CF Card Master/Slave Selection Jumper (JP4)

Pin	Definition
1-2	Master (default)
2-3	Slave

▶ GPIO Pin Power Selection Jumper (JP6)

Pin	Definition
1-2	3.3V (default)
2-3	5V Standby

2.6 Internal Connectors

VGA Connector

Pin	Definition	Pin	Definition
1	RED	2	GREEN
3	BLUE	4	NC
5	Ground	6	Ground
7	Ground	8	Ground
9	VGAVCC	10	Ground
11	NC	12	VCDA
13	HSYNC	14	VSYNC
15	VCLK	16	KEY

Parallel Connector

Pin	Definition	Pin	Definition
1	STB	2	AFD
3	PD0	4	ERR
5	PD1	6	INIT
7	PD2	8	SLIN
9	PD3	10	GND
11	PD4	12	GND
13	PD5	14	GND
15	PD6	16	GND
17	PD7	18	GND
19	ACK	20	GND
21	BUSY	22	GND
23	PE	24	GND
25	SECT	26	NC

Front Panel Connector

Pin	Definition	Pin	Definition
1	NC	2	Power LED+
3	HDD LED+	4	Power LED+
5	HDD LED-	6	Power LED-
7	Reset Switch	8	Power Switch
9	Reset Switch	10	Power Switch
11	NC	12	KEY

SMBUS Connector

Pin	Definition	Pin	Definition
1	Ground	2	5V Power
3	SMBCLK	4	SMBDAT
5	12V Power	6	KEY

USB2-3 / USB4-5

Pin	Definition	Pin	Definition
1	5V	2	5V
3	USB D-	4	USB D-
5	USB D+	6	USB D+
7	Ground	8	Ground
9	Key	10	NC

SPI Connector

Pin	Definition	Pin	Definition
1	3V_standby	2	Ground
3	CS	4	CK
5	SO	6	SI
7	HOLOD-	8	KEY

SATA Connector

Pin	Definition
1	Ground
2	TX+
3	TX-
4	Ground
5	RX-
6	RX+
7	Ground

Chassis Intrusion Connector

Pin	Definition	Pin	Definition
1	Caseopen-	2	GND

CPU Fan/ System Fan

Pin	Definition	Pin	Definition
1	Ground	2	12V
3	Sense	4	Speed Control

8+8 GPIO Pin Connector

Pin	Definition	Pin	Definition
1	GPIOA0	2	GPIOB0
3	GPIOA1	4	GPIOB1
5	GPIOA2	6	GPIOB2
7	GPIOA3	8	GPIOB3
9	Power (JP to 3.3V or 5V)	10	Ground
11	KEY	12	Ground
13	GPIOA4	14	GPIOB4
15	GPIOA5	16	GPIOB5
17	GPIOA6	18	GPIOB6
19	GPIOA7	20	GPIOB7

LAN CPLD JTAG Connector

Pin	Definition
1	Power (1.8V)
2	TDO
3	TDI
4	NC
5	NC
6	TMS
7	Ground
8	TCK

Keyboard/Mouse Connector

(This connector is found as J1 on IO module)

Pin	Definition	Pin	Definition
1	MSDATA	2	KBDATA
3	MSCLK	4	KBCLK
5	Ground	6	Ground
7	Key	8	Ground
9	KBMS POWER	10	KBMS POWER

COM2 Connector

(This connector is found as J2 on IO module)

Pin	Definition	Pin	Definition
1	DCD-	2	SIN-
3	SOUT-	4	DTR-
5	Ground	6	DSR-
7	RTS-	8	CTS-
9	RI-	10	Key

Chapter 3

Operation

3.1 System Memory

BNX-C206 has Intel® C206 chipset built-in, and supports dual channel un-buffered DDR3-1066/1333/1600MHz memory modules. Four DIMM slots support up to 32GB Memory Capacity.

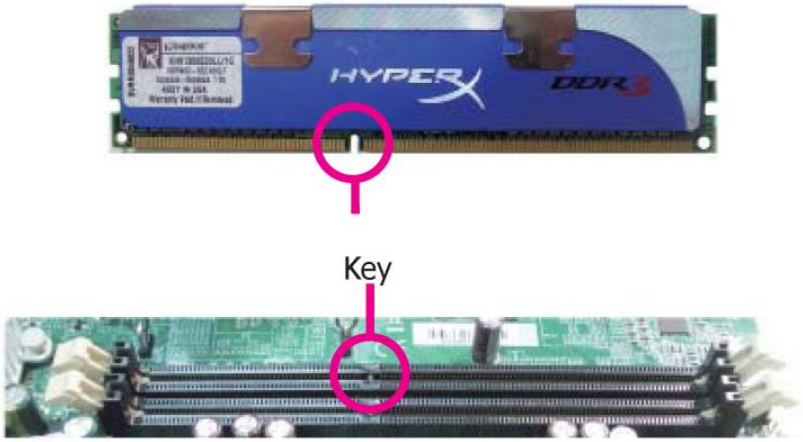
With Intel® Desktop processor, only non-ECC RAM module is supported.
With Intel® E3-1200 or E3-1200 V2 Xeon processor, both non-ECC and ECC RAM module are supported.

3.2 Installing Memory

To install Memory

1. Open up the two side tabs on the slot.

2. Make sure the “Key” on Memory module and socket are perfectly matched, and add slowly the RAM module into the socket along the plastic guides at both ends.



3. Make sure the Memory modules are perfectly snapped by the socket, and the two tabs are restored back to “close” position automatically. If not, press the tabs to “close” position.

4. To remove the Memory modules, please push the tabs outwards, and the memory modules will be automatically disengaged.

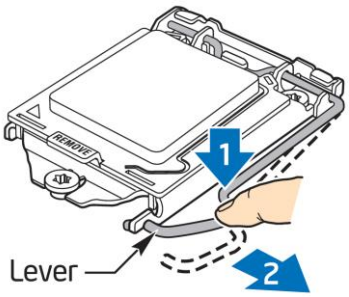
3.3 Installing LGA1155 Intel® Core-i CPU, Heatsink, and Fan

BNX-C206 supports Intel® LGA1155 Gen-2/3 Core-i Processor or E3-1200 (V2) Processor. For a reference list of supported processor, please refer to the specification section. The socket 1155 is formed up with sensitive arrays of pins, improper or careless installation may cause permanent harm to the socket pins. In some cases users may accidentally damage the socket simply by adjusting the position of the CPU.

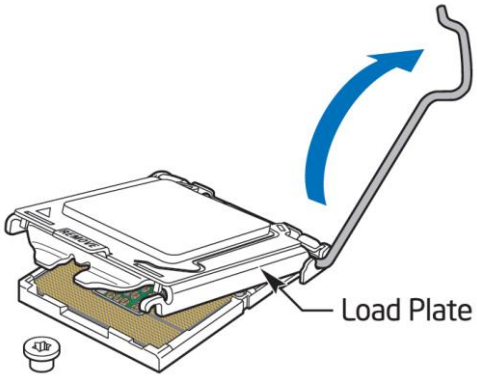
Please follow the installation instructions as shown below:

Step (A): Opening the Socket

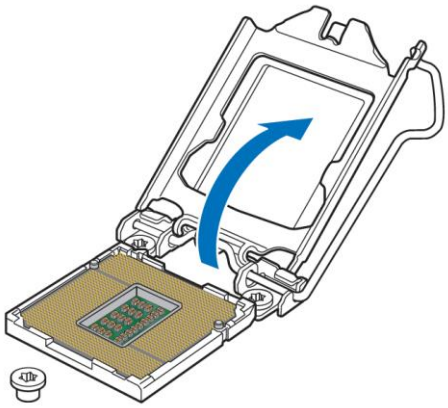
1. Push the "Socket Lever" down and away from the socket to release it.



2. Rotate the "Socket Lever" to lift the "Load Plate" away from the socket.



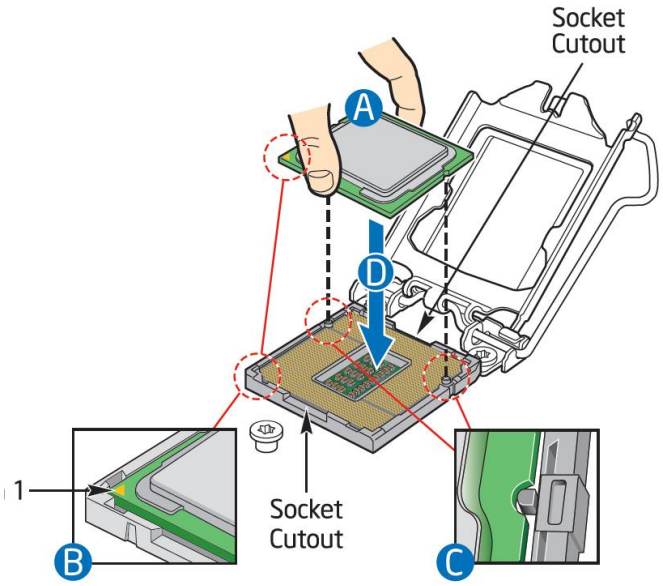
3. Make sure the "Load Plate" is in the fully open position, and also remove the socket cover.



4. When opening the socket, DO NOT TOUCH the gold socket contacts.

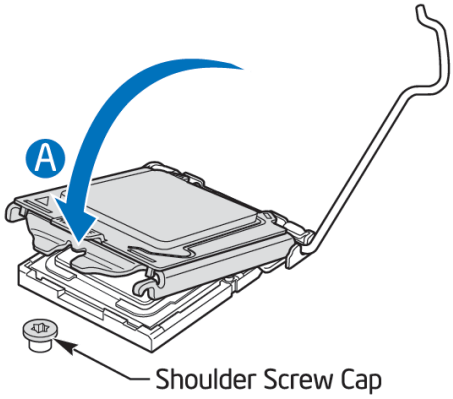
Step (B): Install the Processor

1. Hold the processor with your thumb and index finger as shown to align your fingers with the socket cutouts.
2. Make sure that the processor Pin#1 indicator (gold triangle) is aligned with the Pin#1 chamfer on the socket.
3. Make sure that the notches on the processor align with the posts on the socket.
4. Lower the processor straight down without tilting or sliding it in the socket.

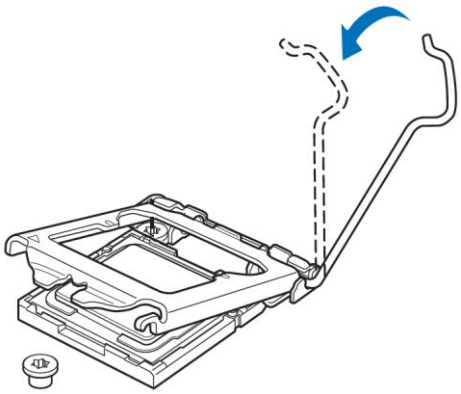


Step (C): Close the Load Plate

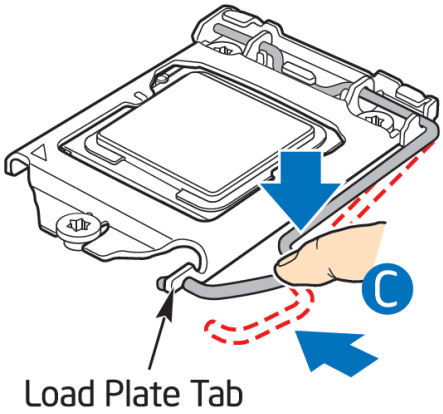
1. Carefully lower the "Load Plate" and make sure it slides under the shoulder screw cap as the lever is lowered.



2. Continue to low the lever.



3. Latch the socket lever under the load plate tab.



Step (D): Install a Heatsink

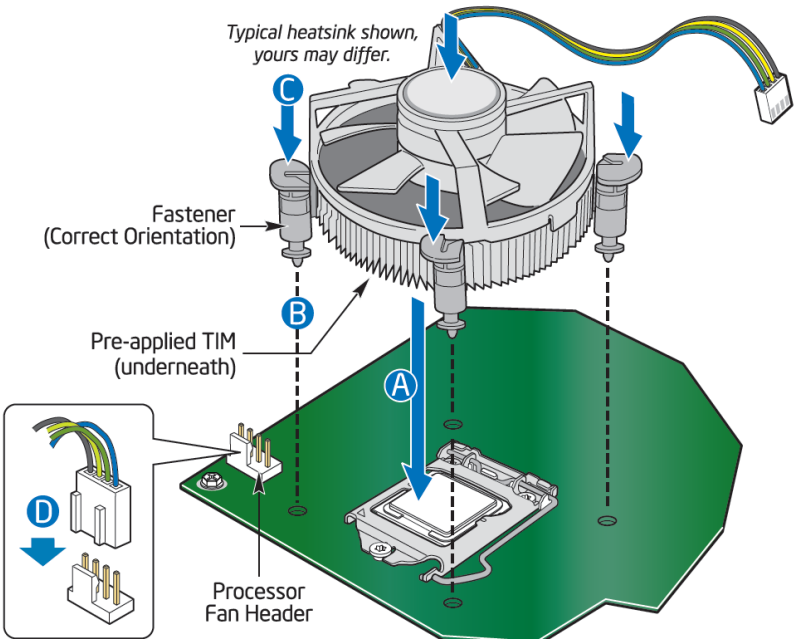
Note: Heatsinks that come with boxed Intel® processors use pre-applied Thermal Interface Material (TIM) and do not need thermal grease. If a different heatsink is used, please refer to manufacturer's instructions.

1. Place the heatsink onto the processor socket. Ensure that the fan power cable is on the side closet to the processor fan header.

2. Align the fasteners with the corresponding board holes (four of them). Ensure that the fasteners slots are pointing perpendicular to the heatsink.

3. While pressing down on the heatsink, press down on the top of the fasteners with your thumb to lock into place. Ensure that all four fasteners are secured.

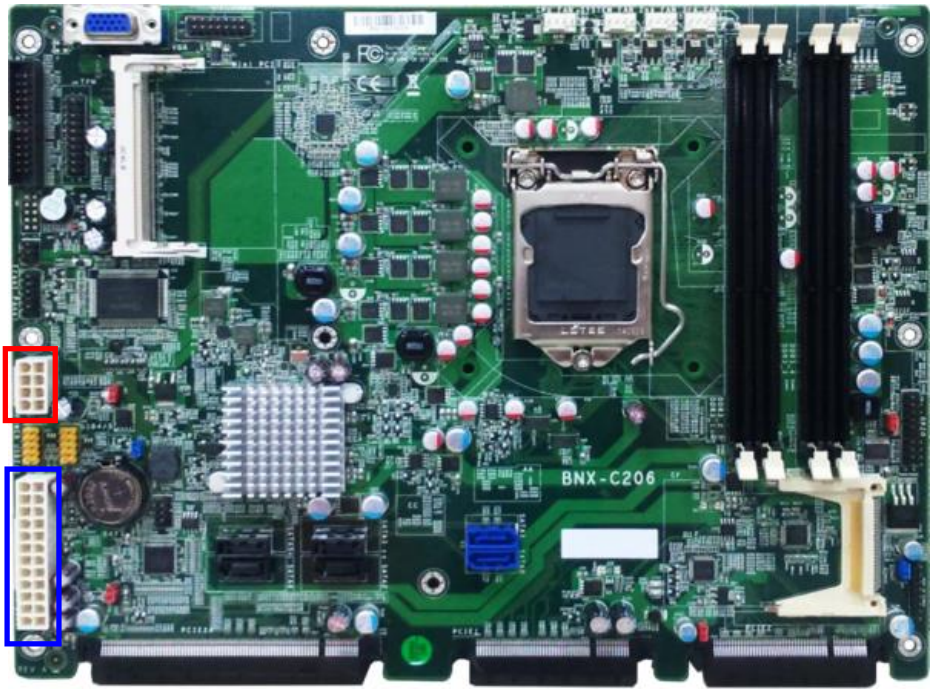
4. Connect the heatsink fan power cable to the processor fan header.



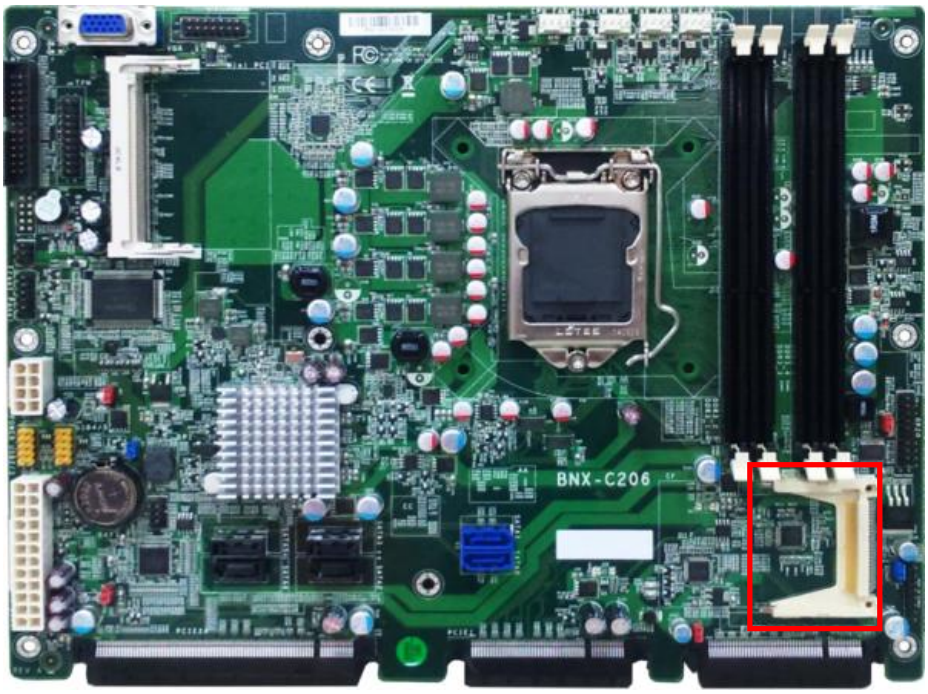
3.4 Adding Power Connectors

BNX-C206 motherboard requires correct power plan to properly support Intel® LGA1155 Processor. Please add the 24-pin ATX Power Connector on the blue connector, and 8-pin 12Vdc Power Connector on the red connector. In case your power supply does not come with 8-pin 12Vdc Power Connector, please at least add the 4-pin 12Vdc Power Connector onto it.

In case the 12Vdc Power Connector is not added, power supply will be triggered, but the motherboard would not boot at all. Please shutdown the system, and add the 12Vdc Power Connector back on.

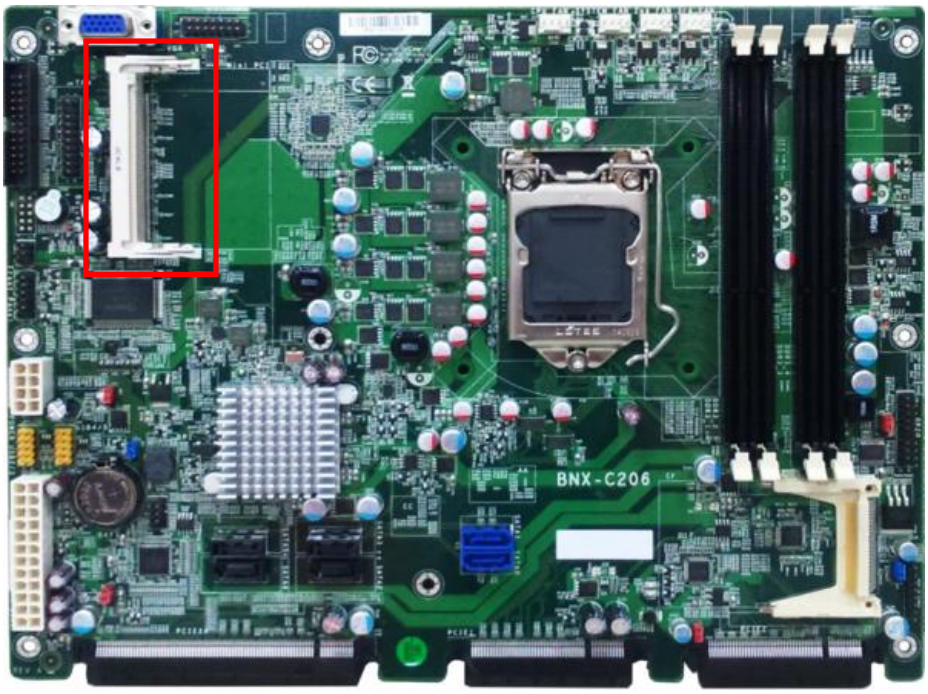


3.5 Add CompactFlash Card



- Please do not add CFast card onto this socket.
- Make sure to shut down the system board before adding your CompactFlash card.

3.6 Add miniPCI Card



- Make sure to shut down the system board before adding your miniPCI card.

Chapter 4

BIOS Setup

About the BIOS

The BIOS (Basic Input and Output System) Setup program is a menu driven utility that enables you to make changes to the system configuration and tailor your system to suit your individual work needs. It is a ROM-based configuration utility that displays the system's configuration status and provides you with a tool to set system parameters. These parameters are stored in non-volatile battery-backed-up CMOS RAM that saves this information even when the power is turned off. When the system is turned back on, the system is configured with the values stored in CMOS.

With easy-to-use pull down menus, you can configure such items as:

- Hard drives, diskette drives, and peripherals
- Video display type and display options
- Password protection from unauthorized use
- Power management features

When to Run BIOS

This program should be executed under the following conditions:

- When changing the system configurations.
- When a configuration error is detected by the system and you are prompted to make changes to the Setup program.
- When resetting the system clock.
- When setting the CPU clock speed so that it automatically runs either fast or slow.
- When redefining the communication ports to prevent any conflicts.
- When making changes to the Power Management configuration.
- When changing the password or making other changes to the security setup.

Normally, CMOS setup is needed when the system hardware is not consistent with the information contained in the CMOS RAM, whenever the CMOS RAM loses power, or when the system features need to be changed.

When to Update BIOS

In the event that new features are released and a BIOS update is required, you will need to update your BIOS on your own, with the help of an appropriate guide, a reference tool, and some command files for the job.

Please seek for help from your local dealer, or send your request to our technical support department.

4.1 Entering Setup

When the system is powered on, the BIOS will initiate the Power-On-Self-Test (POST) routines. These routines perform various diagnostic checks. If an error is encountered, the error will be reported in one of two different ways:

- If the error occurs before the display device is initialized, a series of beeps will be transmitted.
- If the error occurs after the display device is initialized, the screen will display the error message.

Powering on the computer and immediately pressing <**Del**> allows you to enter Setup. Another way to enter Setup is to power on the computer and wait for the following message during the POST:

TO ENTER SETUP BEFORE BOOT
PRESS <CTRL-ALT-ESC> OR <DEL> KEY

Press the <**Del**> key or press the <**Ctrl**>, <**Alt**>, and <**Esc**> keys to enter Setup.

4.2 Getting Help

The online description of the highlighted setup item is displayed at the right pane of the menu at all time. Press F1 to pop up a small help window that lists all the function keys and its use.

To exit the Help Window, press **<F1>** or **<Esc>.**

4.3 Control Keys

The table below lists all the function keys for the navigation in the BIOS setup menu.

Function Key	**Description**
Up/Down Arrow Key	Move Up/Down
Left/Right Arrow Key	Move Left/Right
Enter Key	Select
+/- Key	Change value
ESC	Exit
F1	General Help
F2	Previous Values
F3	Optimized Vales
F4	Save & Exit

To exit the Help Window, press **<F1>** or **<Esc>.**

4.4 The Main Menu

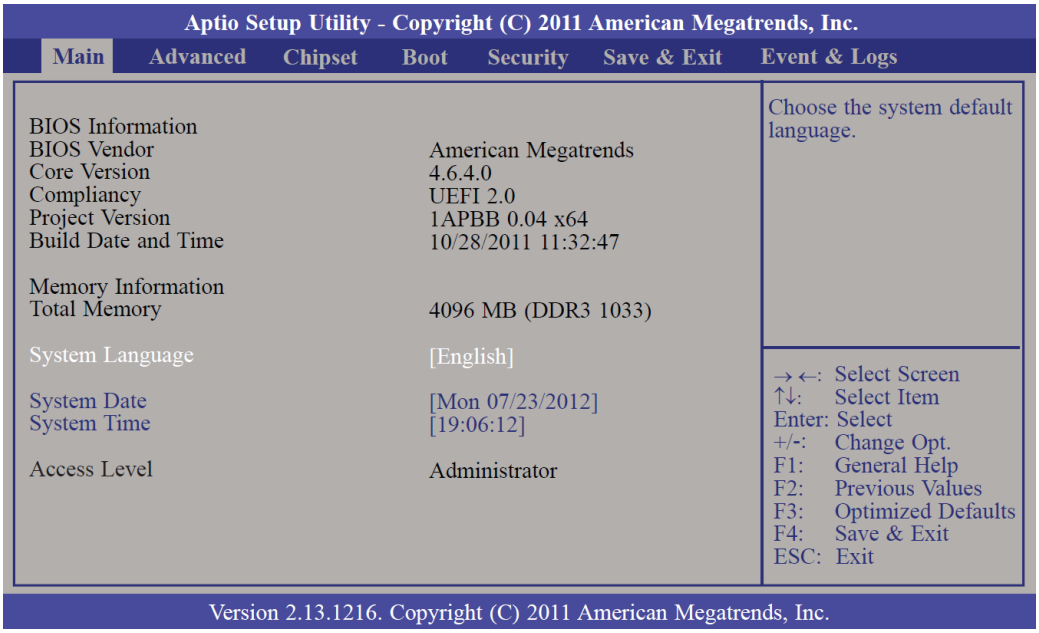


Figure 4-1: BIOS Setup Utility Main Menu

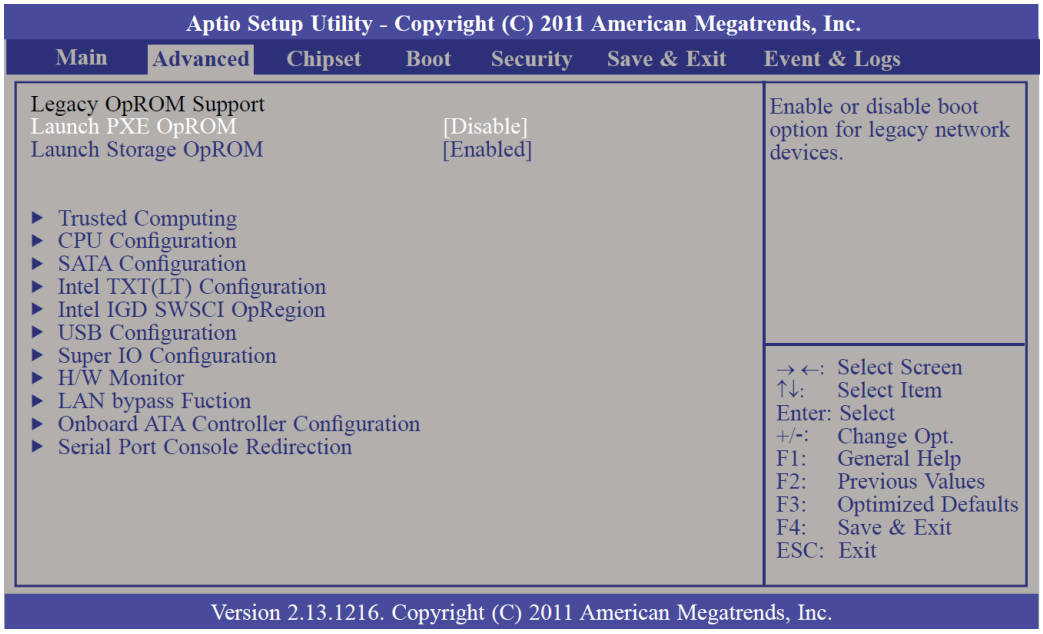
The menu bar on the top of the first screen has the following submenus:

- **Main:** Basic system configuration.
- **Advanced:** Advanced system settings.
- **Chipset:** Other functions.
- **Boot:** System boot configuration.
- **Security:** Configure Supervisor and User Password
- **Save & Exit:** Exit options as well as loading optimal defaults
- **Event & Logs:** Store system events, such as chassis intrusion and power loss, etc.

System Time [xx:xx:xx]: Set the system time.
System Date [Day xx/xx/xxxx]: Set the system date.

4.5 The Advanced Menu

In this section, you may set the configurations for the following items: Trusted Computing, CPU Configuration, SATA Configuration, USB Configuration, Super IO Configuration, H/W Monitor, LAN Bypass Function, and Serial Port Console Redirection.



Launch PXE OpROM

Enable or disable boot options for legacy network devices.

Launch Storage OpROM

Enable or disable boot option for legacy mass storage devices with OpROM

CPU Configuration

Limit CPUID Maximum

It is designed to limit the listed speed of the processor to older operating systems.

SATA Configuration

SATA Mode

- *IDE Mode:* This option configures the Serial ATA drives as Parallel ATA storage devices.
- *AHCI Mode:* This option allows the Serial ATA devices to use AHCI (Advanced Host Controller Interface).
- *RAID Mode:* This option allows you to create RAID or Intel Matrix Storage configuration on Serial ATA devices.

Serial ATA Controller 0-1

This option allows you to Enable/Disabled Serial ATA Controller 0-1.

If AHCI or RAID mode is selected, all detected SATA Hard Drives will be prompted on the list.

USB Configuration

Legacy USB Support

- *Enabled:* Enables legacy USB.
- *Auto:* Disables support for legacy when no USB devices are connected.
- *Disabled: Keep USB devices available only for EFI applications.*

EHCI Hand-off

This is a workaround for OSes that does not support EHCI hand-off. The EHCI ownership change should be claimed by the EHCI driver.

Port 60/64 Emulation

Enables I/O port 60h/64h emulation support. This should be enabled for the complete USB keyboard legacy support for non-USB aware OSes.

Super IO Configuration

Restore AC Power Loss

- *Off:* When power returns after an AC power failure, the system's power is off. You must press the Power button to power-on the system.
- *On:* When power returns after an AC power failure, the system will automatically power-on.
- *Last State:* When power returns after an AC power failure, the system will return to the state where you left off before power failure occurs. If the system's power is off when AC power failure occurs, it will remain off when power returns. If the system's power is on when AC power failure occurs, the system will power-on when power returns.

WatchDog Timer

This is used to select the time interval of the Watchdog timer. If the system hangs or fails to function, it will reset at the set time interval so that your system will continue to operate.

Serial Port 0/1 Configuration

Serial Port

Enables or disables the serial port.

Change Settings

Selects the IO/IRQ setting of the I/O device.

Parallel Port Configuration

Parallel Port

Enables or disables the Parallel port.

Change Settings

Selects the IO/IRQ setting of the I/O device.

Device Mode

Change the printer port mode.

LAN Bypass Function

LAN Group1 Power ON and LAN Group2 Power ON

The options are Pass Through and Bypass.

LAN Group1 Power OFF and LAN Group2 Power OFF

The options are Pass Through and Bypass.

WDT

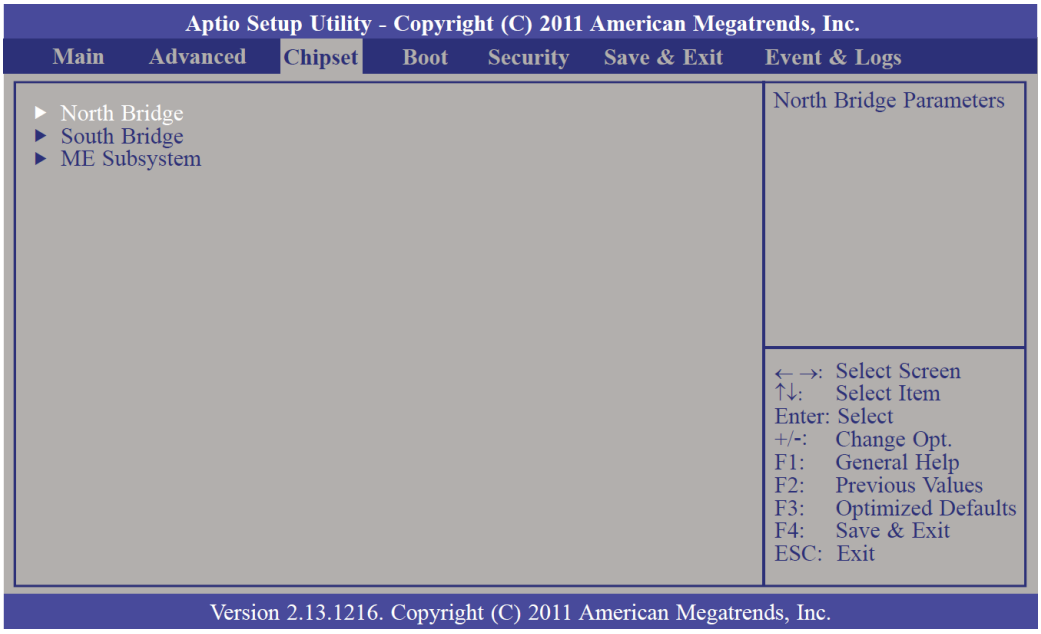
This is used to enable or disable the Watchdog Timer function.

Serial Port Console Redirection

Console Redirection

Enables or disables Console Redirection function of each Serial Port.

4.6 The Chipset Menu



North Bridge

IGD Memory

Selects the internal graphics device's shared memory size.

South Bridge

Wake on LAN from S5

When enabled, it allows the system to wake from S5 via the network port.

After G3

Power Off / WOL

Power-on the system via WOL after G3.

Power On

Power-on the system after G3.

USB Configuration

All USB Devices

Enables or disables all USB devices.

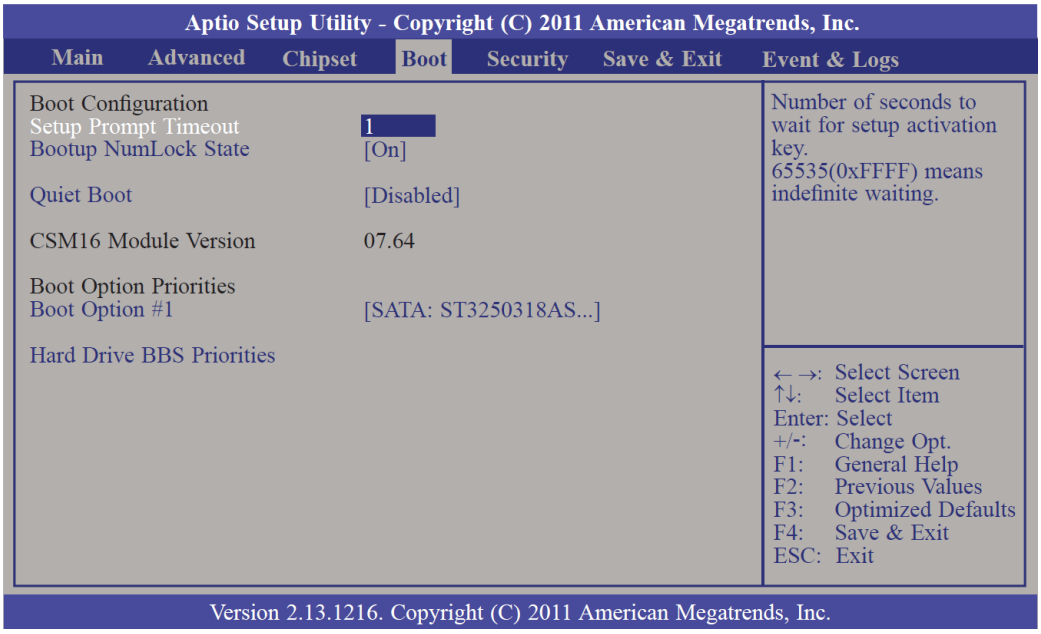
EHCI Controller 1 and EHCI Controller 2

These are used to enable or disable Enhanced Host Controller Interface (USB 2.0).

USB Port 0 to USB Port 5

Enables or disables the selected USB port.

4.7 The Boot Menu



Bootup NumLock State

This setting is to set the Num Lock status when the system is powered on. Setting to [On] will turn on the Num Lock key when the system is powered on. Setting to [Off] will allow users to use the arrow keys on the numeric keypad.

Quiet Boot

This BIOS feature determines if the BIOS should hide the normal POST messages with the motherboard or system manufacturer's full-screen logo.

- When it is enabled, the BIOS will display the full-screen logo during the boot-up sequence, hiding normal POST messages.
- When it is disabled, the BIOS will display the normal POST messages, instead of the full-screen logo.

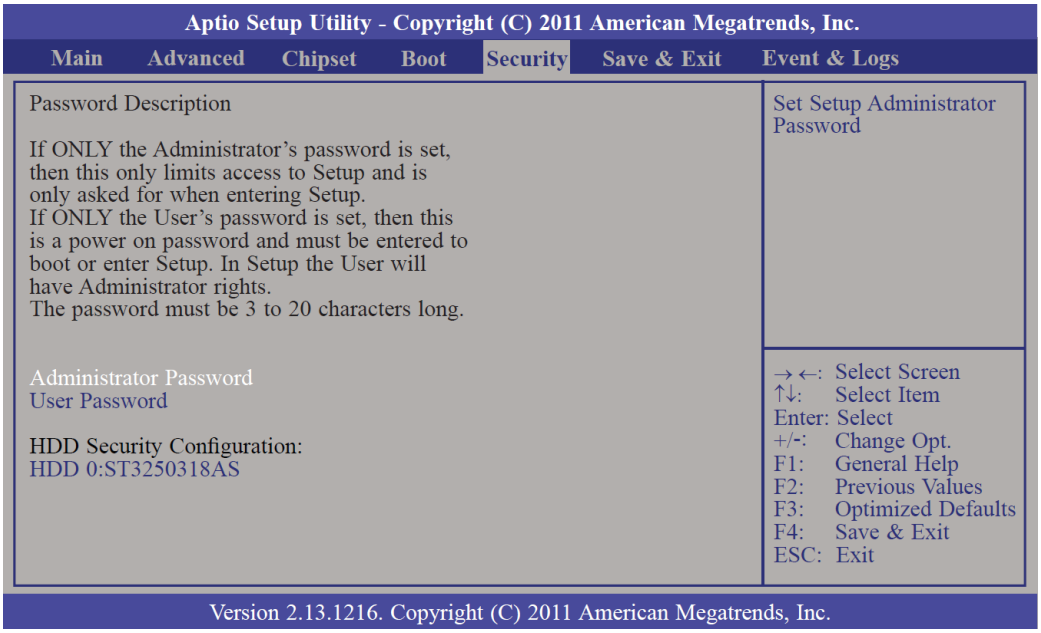
Boot Option Priorities

The items allow you to set the sequence of boot devices where BIOS attempts to load the disk operating system. First press <Enter> to enter the sub-menu. Then you may use the arrow keys (↑↓) to select the desired device, then press <+>, <-> or <PageUp>, <PageDown> key to move it up/down in the priority list.

Hard Drive BBS Priorities

This setting allows users to set the Hard Drive boot sequence.

4.8 The Security Menu


Aptio Setup Utility - Copyright (C) 2011 American Megatrends, Inc.
Main
Advanced
Chipset
Boot
Security
Save & Exit
Event & Logs
Password Description
If ONLY the Administrator's password is set, then this only limits access to Setup and is only asked for when entering Setup.
If ONLY the User's password is set, then this is a power on password and must be entered to boot or enter Setup. In Setup the User will have Administrator rights.
The password must be 3 to 20 characters long.
Administrator Password
User Password
HDD Security Configuration:
HDD 0:ST3250318AS
Set Setup Administrator Password
→←: Select Screen
↑↓: Select Item
Enter: Select
+/-: Change Opt.
F1: General Help
F2: Previous Values
F3: Optimized Defaults
F4: Save & Exit
ESC: Exit
Version 2.13.1216. Copyright (C) 2011 American Megatrends, Inc.

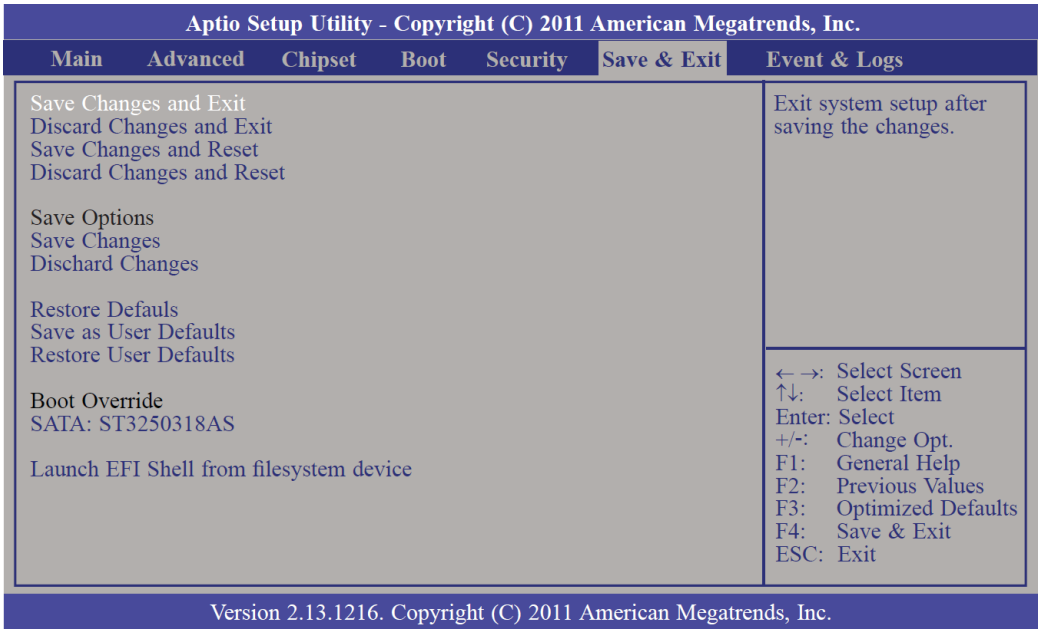
Administrator Password

Administrator Password controls access to the BIOS Setup utility. Users will be prompted for Administrator password only when they enter BIOS Setup.

User Password

User Password controls access to the system at boot and access to the BIOS Setup utility. Users will be prompted for User password when they power on the system or enter BIOS Setup. In BIOS Setup, users will have Administrator rights.

4.9 The Save & Exit Menu



Discard Changes and Exit

To exit the Setup utility without saving the changes, select this field then press <Enter>. You may be prompted to confirm again before exiting.

You can also press <ESC> to exit without saving the changes.

Save Changes and Reset

To save the changes, select this field and then press <Enter>. A dialog box will appear. Select Yes to reset the system after saving all changes made.

Discard Changes and Reset

To discard the changes, select this field and then press <Enter>. A dialog box will appear. Select Yes to reset the system setup without saving any changes.

Save Changes

Save changes done so far to any of the setup options.

Dischard Changes

To discard the changes, select this field then press <Enter>. A dialog box will appear. Confirm by selecting Yes to discard all changes made and restore the previously saved settings.

You can also press <F7> to discard the changes.

Restore Defaults

To restore and load the optimized default values, select this field and then press <Enter>. A dialog box will appear. Select Yes to restore the default values of all the setup options.

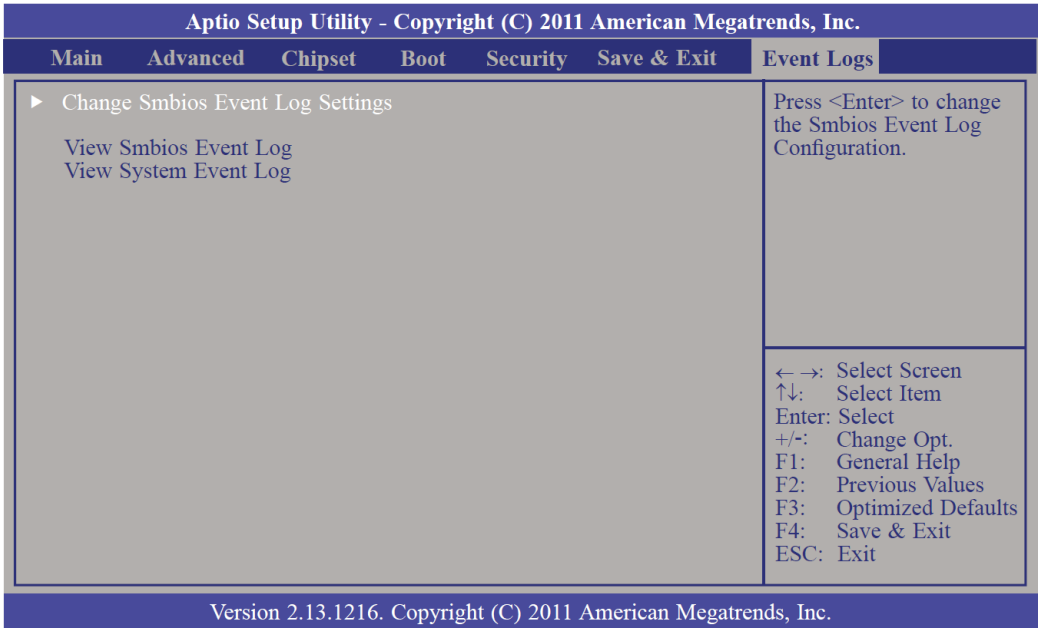
Save as User Defaults

To save changes done so far as user default, select this field and then press <Enter>. A dialog box will appear. Select Yes to save values as user default.

Restore User Defaults

To restore user default to all the setup options, select this field and then press <Enter>. A dialog box will appear. Select Yes to restore user default.

4.10 Event Logs



View Smbios Event Log

To view Smbios event log.

View System Event Log

To view system event log.

Change Smbios Event Log Settings

Erase Event Log

Choose options for erasing Smbios event log. Erasing is done prior to any logging activation during reset.

When Log is Full

Choose options for reactions to a full Smbios event log.

MECI

Multi Event Count Increment: The number of occurrences of a duplicate event that must pass before the multiple-event counter associated with the log entry is updated, specified as a numeric value in the range of 1 to 255.

METW

Multi Event Time Window: The number of minutes which must pass between duplicate log entries which utilize a multiple-event counter. The value ranges from 0 to 99.

Log OEM Codes

Enable or disable the logging of EFI Status codes as OEM codes (if not already converted to legacy)

Convert OEM Codes

Enable or disable the converting of EFI Status Codes to Standard Smbios Types (Not all may be translated).

Chapter 5

Programming Guide

5.1 LAN Status Sample Code

```
#include <stdio.h>

//---------------------------------------------------------------------

//CMOS Port Address
#define CMOS_INDX_PORT    0x70
#define CMOS_DATA_PORT    0x71

//LAN Status Flag Address
#define  LAN_STATUS_FLAG1   Ox55
#define  LAN_STATUS_FLAG2   Ox56

//LAN Status Flag Definition
#define LAN1_PWRON_FLAG    0x00
#define LAN1_PWROFF_FLAG   0x01
#define LAN2_PWRON_FLAG    0x02
#define LAN2_PWROFF_FLAG   0x03
#define LAN_WDT_FLAG       0x04
#define LAN3_PWRON_FLAG    0x05
#define LAN3_PWROFF_FLAG   0x06
#define LAN4_PWRON_FLAG    0x00
#define LAN4_PWROFF_FLAG   0x01
//---------------------------------------------------------------------
void GpioInit(void);
void GpioOut(char, char);
void SetBit(char*, char);
void ClrBit(char*, char);
char GetBit(char Data, char Bit);
//---------------------------------------------------------------------
main(void)
{
char data1, data2;
char  *BP  = "ByPass";
char  *PT  = "PassThrough";
char  *RST = "Reset";
```

```
char LAN1PowerON[15], LAN1PowerOFF[15], LAN2PowerON[15], LAN2PowerOFF[15];
char LAN3PowerON[15], LAN3PowerOFF[15], LAN4PowerON[15], LAN4PowerOFF[15];
char WDT[15];

outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG1);
data1 = inportb(CMOS_DATA_PORT);
outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG2);
data2 = inportb(CMOS_DATA_PORT);
if(GetBit(data1, LAN1_PWRON_FLAG))        strcpy(Lan1PowerON, BP);
else                                      strcpy(Lan1PowerON, PT);
if(GetBit(data1, LAN1_PWROFF_FLAG))       strcpy(Lan1PowerOFF, BP);
else                                      strcpy(Lan1PowerOFF, PT);
if(GetBit(data1, LAN2_PWRON_FLAG))        strcpy(Lan2PowerON, BP);
else                                      strcpy(Lan2PowerON, PT);
if(GetBit(data1, LAN2_PWROFF_FLAG))       strcpy(Lan2PowerOFF, BP);
else                                      strcpy(Lan2PowerOFF, PT);
if(GetBit(data1, LAN_WDT_FLAG))           strcpy(WDT, BP);
else                                      strcpy(WDT, Rst);
if(GetBit(data1, LAN3_PWRON_FLAG))        strcpy(Lan3PowerON, BP);
else                                      strcpy(Lan3PowerON, PT);
if(GetBit(data1, LAN3_PWROFF_FLAG))       strcpy(Lan3PowerOFF, BP);
else                                      strcpy(Lan3PowerOFF, PT);
if(GetBit(data2, LAN4_PWRON_FLAG))        strcpy(Lan4PowerON, BP);
else                                      strcpy(Lan4PowerON, PT);
if(GetBit(data2, LAN4_PWROFF_FLAG))       strcpy(Lan4PowerOFF, BP);
else                                      strcpy(Lan4PowerOFF, PT);

printf("[LAN 1]\n")
printf("POWER ON:%s\n", LAN1PowerON);
printf("POWER OFF:%s\n", LAN1PowerOFF);
printf("[LAN 2]\n")
printf("POWER ON:%s\n", LAN2PowerON);
printf("POWER OFF:%s\n", LAN2PowerOFF);
printf("[LAN 3]\n")
printf("POWER ON:%s\n", LAN3PowerON);
printf("POWER OFF:%s\n", LAN3PowerOFF);
printf("[LAN 4]\n")
printf("POWER ON:%s\n", LAN4PowerON);
```

```
printf("POWER OFF:%s\n", LAN4PowerOFF);
printf("[WDT]\n")
printf("WDT Event:%s\n", WDT);

return;
}
//---------------------------------------------------------------------
void SetBit(char *Data, char Bit)
{
switch(Bit)
{
    switch(Bit)
    {
        case 0:
                *Data |= 0x01;
                break;
        case 1:
                *Data |= 0x02;
                break;
        case 2:
                *Data |= 0x04;
                break;
        case 3:
                *Data |= 0x08;
                break;
        case 4:
                *Data |= 0x10;
                break;
        case 5:
                *Data |= 0x20;
                break;
        case 6:
                *Data |= 0x40;
                break;
        case 7:
                *Data |= 0x80;
                break;
    }
}
```

```
void ClrBit(char *Data, char Bit)
{
    switch(Bit)
    {

        case 0:
                  *Data &= 0xFE;
                  break;
        case 1:
                  *Data &= 0xFD;
                  break;
        case 2:
                  *Data &= 0xFB;
                  break;
        case 3:
                  *Data &= 0xF7;
                  break;
        case 4:
                  *Data &= 0xEF;
                  break;
        case 5:
                  *Data &= 0xDF;
                  break;
        case 6:
                  *Data &= 0xBF;
                  break;
        case 7:
                  *Data &= 0x7F;
                  break;
    }
}

//---------------------------------------------------------------------------

char GetBit(char Data, char Bit)
{
    switch(Bit)
    {
```

```
        case 0:
                    *Data &= 0x01;
                    break;
        case 1:
                    *Data &= 0x02;
                    break;
        case 2:
                    *Data &= 0x04;
                    break;
        case 3:
                    *Data &= 0x08;
                    break;
        case 4:
                    *Data &= 0x10;
                    break;
        case 5:
                    *Data &= 0x20;
                    break;
        case 6:
                    *Data &= 0x40;
                    break;
        case 7:
                    *Data &= 0x80;
                    break;
    }
    if (data == 0)   return 0;
    else             return 1;
}
//---------------------------------------------------------------------------
```

5.2 LAN Switch Sample Code

```
#include <stdio.h>

//---------------------------------------------------------------------------

//LPC Port Address
#define ADDR_PORT         0x2E
#define DATA_PORT         0x2F

//GPIO Pin Definition
#define LAN_LOCK          0x04
#define LAN_COMMAND       0x07
#define LAN_PWRON         0x01
#define LAN_PWROFF        0x05
#define LAN_WDT           0x06

//CMOS Port Address
#define CMOS_INDX_PORT    0x70
#define CMOS_DATA_PORT    0x71

//LAN Status Flag Address
#define LAN_STATUS_FLAG1  0x55
#define LAN_STATUS_FLAG2  0x56

//LAN Status Flag Definition
#define LAN1_PWRON_FLAG   0x00
#define LAN1_PWROFF_FLAG  0x01
#define LAN2_PWRON_FLAG   0x02
#define LAN2_PWROFF_FLAG  0x03
#define LAN_WDT_FLAG      0x04
#define LAN3_PWRON_FLAG   0x055
#define LAN3_PWROFF_FLAG  0x06
#define LAN4_PWRON_FLAG   0x00
#define LAN4_PWROFF_FLAG  0x01
//---------------------------------------------------------------------------
void GpioInit(void);
void GpioOut(char, char);
```

```
void SetBit(char*, char);
void ClrBit(char*, char);
void SetStatusFlag(char, char, char, char);
//---------------------------------------------------------------------------
main(int argc, char *argv[])
{
int PairSel, PowerOn, PowerOff, WDT;

if(argc == 5)
{
       PairSel    = strtol(argv[1], NULL, 16);
       PowerOn    = strtol(argv[2], NULL, 16);
       PowerOff   = strtol(argv[3], NULL, 16);
       WDT        = strtol(argv[4], NULL, 16);
}
else
{
       printf("Command Error!!\n");
       return;
}
//Init GPIO
GpioInit();
GpioOut(LAN_LOCK, 0);
GpioOut(LAN_COMMAND, 0);

//Command Start: Set LAN_LOCK pin to high level
GpioOut(LAN_LOCK, 1);
delay(1);

//ByPass/PassThough Setting
//Set High/Low level to corresponding pins
// High => Bypass
// Low => PassThrough
GpioOut(LAN_PWRON, PowerOn);
GpioOut(LAN_PWROFF, PowerOff);
GpioOut(LAN_WDT, WDT);

//LAN Pair Selection
//Generate N pulses to select the Nth LAN pair
```

```
switch(PairSel)
{
    case 1:   GpioOut(LAN_COMMAND, 1);//            ____
              delay(1);//                        __|1ms|__
              break;

    case 2:   GpioOut(LAN_COMMAND, 1);
              delay(1);
              GpioOut(LAN_COMMAND, 0);
              delay(1);
              GpioOut(LAN_COMMAND, 1);//            ____   ___
              delay(1);//                        __|1ms|__|1ms|__
              break;

    case 3:   GpioOut(LAN_COMMAND, 1);
              delay(1);
              GpioOut(LAN_COMMAND, 0);
              delay(1);
              GpioOut(LAN_COMMAND, 1);
              delay(1);
              GpioOut(LAN_COMMAND, 0);
              delay(1);
              GpioOut(LAN_COMMAND, 1);//     ____   ___   ____
              delay(1);//                  __|1ms|__|1ms|__|1ms|__
              break;

    case 4:   GpioOut(LAN_COMMAND, 1);
              delay(1);
              GpioOut(LAN_COMMAND, 0);
              delay(1);
              GpioOut(LAN_COMMAND, 1);
              delay(1);
              GpioOut(LAN_COMMAND, 0);
              delay(1);
              GpioOut(LAN_COMMAND, 1);
              delay(1);
              GpioOut(LAN_COMMAND, 0);
              delay(1);
```

```
            GpioOut(LAN_COMMAND, 1);//      ____    ___    ____    ____
            delay(1);//                  __|1ms|__|1ms|__|1ms|__|1ms|__
            break;
}
//Command End: Set LAN_COMMAND and LAN_LOCK to low level
GpioOut(LAN_COMMAND, 0);
GpioOut(LAN_LOCK, 0);

//Save LAN Status to CMOS
SetStatusFlag(PairSel, PowerOn, PowerOff, WDT);

Return;
}
//---------------------------------------------------------------------------
void GpioInit(void)
{
char data;

outportb(ADDR_PORT, 0x87);
outportb(ADDR_PORT, 0x87);

//Select logical device 9
outportb(ADDR_PORT, 0x07);
outportb(DATA_PORT, 0x09);

//Enable GPIO_3 (Set bit 1=1)
outportb(ADDR_PORT, 0x30);
data = inportb(DATA_Port);
data = data | 0x02;
outportb(DATA_PORT, DATA);

//Disable SDA input, Disable SCL input (GP32.33.34) (Set bit1=0, bit2=1,
bit3=1)
outportb(ADDR_PORT, 0x2A);
data = inportb(DATA_Port);
data = data & 0xFD;
data = data | 0x0C;
outportb(DATA_PORT, DATA);
```

```
//Enable GPIO_3 (GP32.33.34) (Set bit5.6.7=0)
outportb(ADDR_PORT, 0x2C);
data = inportb(DATA_Port);
data = data & 0x1F;
outportb(DATA_PORT, DATA);

//Initial GP0, GP1, GP4, GP5, GP6, GP7 pin status
outportb(ADDR_PORT, 0xF1);
data = inportb(DATA_Port);
data = data & 0x0C;
outportb(DATA_PORT, DATA);

//Set GP0, GP1, GP4, GP5, GP6, GP7 as output mode
outportb(ADDR_PORT, 0xF0);
data = inportb(DATA_Port);
data = data & 0x0C;
outportb(DATA_PORT, DATA);

outportb(ADDR_PORT, 0xAA);
}

//-----------------------------------------------------------------------
void GpioOut(char bit, char level)
{
    Char data;

    outportb(ADDR_PORT, 0x87);
    outportb(ADDR_PORT, 0x87);

    outportb(ADDR_PORT, 0xF1);
    data = inportb(DATA_PORT);
    if(Level == 1){
        SetBit(&data,Bit);
    }
    else {
        ClrBit(&data,Bit);
    }
    outportb(DATA_PORT, DATA);
```

```
    outportb(ADDR_PORT, 0xAA);
}
//---------------------------------------------------------------------------
void SetStatusFlag(char LANPair, char PowerON, char PowerOFF, char WDT)
{
    char data;

    switch(LANPair)
    {
    case 1:
            outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG1);
            data = inportb(CMOS_DATA_PORT);
            if(PowerON)  SetBit(&data, LAN1_PWRON_FLAG);
            else  ClrBit (&data, LAN1_PWRON_FLAG);
            if(PowerOFF) SetBit(&data, LAN1_PWROFF_FLAG);
            else  ClrBit (&data, LAN1_PWROFF_FLAG);
            if(WDT)  SetBit(&data, LAN_WDT_FLAG);
            else  ClrBit (&data, LAN_WDT_FLAG);
            outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG1);
            outportb(CMOS_DATA_PORT, data);
            break;
    case 2:
            outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG1);
            data = inportb(CMOS_DATA_PORT);
            if(PowerON)  SetBit(&data, LAN2_PWRON_FLAG);
            else  ClrBit (&data, LAN2_PWRON_FLAG);
            if(PowerOFF) SetBit(&data, LAN2_PWROFF_FLAG);
            else  ClrBit (&data, LAN2_PWROFF_FLAG);
            if(WDT)  SetBit(&data, LAN_WDT_FLAG);
            else  ClrBit (&data, LAN_WDT_FLAG);
            outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG1);
            outportb(CMOS_DATA_PORT, data);
            break;
    case 3:
            outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG1);
            data = inportb(CMOS_DATA_PORT);
            if(PowerON)  SetBit(&data, LAN3_PWRON_FLAG);
            else   ClrBit (&data, LAN3_PWRON_FLAG);
```

```
            if(PowerOFF) SetBit(&data, LAN3_PWROFF_FLAG);
            else   ClrBit (&data, LAN3_PWROFF_FLAG);
            if(WDT)  SetBit(&data, LAN_WDT_FLAG);
            else  ClrBit (&data, LAN_WDT_FLAG);
            outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG1);
            outportb(CMOS_DATA_PORT, data);
            break;
    case 4:
            outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG2);
            data = inportb(CMOS_DATA_PORT);
            if(PowerON)  SetBit(&data, LAN4_PWRON_FLAG);
            else  ClrBit (&data, LAN4_PWRON_FLAG);
            if(PowerOFF) SetBit(&data, LAN4_PWROFF_FLAG);
            else  ClrBit (&data, LAN4_PWROFF_FLAG);
            if(WDT)  SetBit(&data, LAN_WDT_FLAG);
            else  ClrBit (&data, LAN_WDT_FLAG);
            outportb(CMOS_INDX_PORT, LAN_STATUS_FLAG2);
            outportb(CMOS_DATA_PORT, data);
            break;
    }
}
//---------------------------------------------------------------------------
void SetBit (char *Data, char Bit)
{
    switch(Bit);
    {
          case 0:
                  *Data |= 0x01;
                  break;
          case 1:
                  *Data |= 0x02;
                  break;
          case 2:
                  *Data |= 0x04;
                  break;
          case 3:
                  *Data |= 0x08;
                  break;
          case 4:
```

```
                *Data |= 0x10;
                break;
        case 5:
                *Data |= 0x20;
                break;
        case 6:
                *Data |= 0x40;
                break;
        case 7:
                *Data |= 0x80;
                break;
    }
}
//---------------------------------------------------------------------
void ClrBit (char *Data, char Bit)
{
    switch(Bit);
    {
        case 0:
                *Data &= 0xFE;
                break;
        case 1:
                *Data &= 0xFD;
                break;
        case 2:
                *Data &= 0xFB;
                break;
        case 3:
                *Data &= 0xF7;
                break;
        case 4:
                *Data &= 0xEF;
                break;
        case 5:
                *Data &= 0xDF;
                break;
        case 6:
                *Data &= 0xBF;
                break;
```

```
            case 7:
                    *Data &= 0x7F;
                    break;
    }
}
```

5.3 WDT Sample Code

```
//---------------------------------------------------------------------

#include <stdio.h>

//---------------------------------------------------------------------
#define ADDR_PORT 0x2E
#define DATA_PORT 0x2F
//---------------------------------------------------------------------
void WdtInit(void);
void SetWdtTime(char);
char GetWdtTime(void);
void SetBit(char*, char);
void ClrBit(char*, char);

//---------------------------------------------------------------------
main(int argc, char *argv[])
{
int PairSel, PowerOn, PowerOff, WDT;
char count, data;

if(argc==2)
{
                    count = strtol(argv[1], NULL, 10);
}
else
{
                    printf("Command Error!!\n");
                    return;
}

WdtInit();
SetWdtTime(count);

while(1)
{
                    Data = GetWdtTime();
```

```
                    delay(100);
                    printf("\r Time Remaining: %d", data);
                    if(data==1) break;
}
delay(1000);
printf("\r Time Remaing: 0");
printf("\n Time Up!!\n");

WdtInit();
SetWdtTime(0);
return;
}
//---------------------------------------------------------------------------
void WdtInit(void)
{
char data;

outportb(ADDR_PORT, 0x87);
outportb(ADDR_PORT, 0x87);

//Set WDT Pin77 as WDTO#
outportb(ADDR_PORT, 0x2D);
data = inportb(DATA_PORT);
ClrBit(&data, 0);
outportb(DATA_PORT, data);

//Select LDN 8
outportb(ADDR_PORT, 0x07);
outportb(DATA_PORT, 0x08);

//Clear WDTO# Status bit
outportb(ADDR_PORT, 0xF7);
data = inportb(DATA_PORT);
ClrBit(&data, 4);
outportb(DATA_PORT, data);

//Enable Watch Dog Timer
outportb(ADDR_PORT, 0x30);
data = inportb(DATA_PORT);
```

```
SetBit(&data, 0);
outportb(DATA_PORT, data);

//Select Second Mode
outportb(ADDR_PORT, 0xF5);
data = inportb(DATA_PORT);
ClrBit(&data, 3);
outportb(DATA_PORT, data);

outportb(ADDR_PORT, 0xAA);
}
//---------------------------------------------------------------------
void SetWdtTime(char count)
{
outportb(ADDR_PORT, 0x87);
outportb(ADDR_PORT, 0x87);

//Select LDN 8
outportb(ADDR_PORT, 0x07);
outportb(DATA_PORT, 0x08);

//Set Count
outportb(ADDR_PORT, 0xF6);
outportb(DATA_PORT, count);

outportb(ADDR_PORT, 0xAA);
}
//---------------------------------------------------------------------
char GetWdtTime(void)
{
char data;

outportb(ADDR_PORT, 0x87);
outportb(ADDR_PORT, 0x87);

//Select LDN 8
outportb(ADDR_PORT, 0x07);
outportb(DATA_PORT, 0x08);
```

```
//Get Remaining Count
outportb(ADDR_PORT, 0xF6);
data = inportb(DATA_PORT);

outportb(ADDR_PORT, 0xAA);

return data;
}
//---------------------------------------------------------------------------
void SetBit(char *Data, char Bit)
{
    switch(Bit)
    {
                    case 0:
                              *Data |= 0x01;
                              break;
                    case 1:
                              *Data |= 0x02;
                              break;
                    case 2:
                              *Data |= 0x04;
                              break;
                    case 3:
                              *Data |= 0x08;
                              break;
                    case 4:
                              *Data |= 0x10;
                              break;
                    case 5:
                              *Data |= 0x20;
                              break;
                    case 6:
                              *Data |= 0x40;
                              break;
                    case 7:
                              *Data |= 0x80;
                              break;
    }
}
```

```
//---------------------------------------------------------------------------
void ClrBit(char *Data, char Bit)
{
    switch(Bit)
    {
                    case 0:
                              *Data |= 0xFE;
                              break;
                    case 1:
                              *Data |= 0xFD;
                              break;
                    case 2:
                              *Data |= 0xFB;
                              break;
                    case 3:
                              *Data |= 0xF7;
                              break;
                    case 4:
                              *Data |= 0xEF;
                              break;
                    case 5:
                              *Data |= 0xDF;
                              break;
                    case 6:
                              *Data |= 0xBF;
                              break;
                    case 7:
                              *Data |= 0x7F;
                              break;
    }
}
//---------------------------------------------------------------------------
```